ITB8832 Mathematics for Computer Science
Lecture 2 – 11 September 2022

- Chapter One
  - Proofs by Contradiction
- Chapter Two
  - Well Ordering Proofs
  - Factorization into Primes
  - Well Ordered Sets
- Chapter Three
  - Ambiguity in Human Language
  - Propositions from Propositions
  - Propositional Logic in Computer Science

Contents

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. Propositional Logic in Computer Programs

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. Propositional Logic in Computer Programs

The Law of Non-Contradiction

Law of Non-Contradiction

It is impossible that something and its negation are both true at the same time.

In formula:

$$\frac{}{\text{not}(A \text{ and } \text{not}(A))}$$

This could be the most important principle of logic.

The Law of the Excluded Middle

Law of the Excluded Middle

Given anything and its negation, one of the two is true.

In formula:

$$\frac{}{A \text{ or } \mathrm{not}(A)}$$

This is another fundamental principle of *classical* logic.

- However, there are other logics where the Law of the Excluded Middle is not valid.
- This happens, for example, in *intuitionistic logic*, where a proof of A is a *witness* that A is true.
- In this context, a witness of A implies B is a “black box” that transforms witnesses of A into witnesses of B; that is, a *function* from A to B.
- Then not(A) is *defined* as A implies $\perp$, where $\perp$ (read “bottom”) is a proposition that has no witnesses.
- It is *always* possible to construct a witness of not(not(A)) from a witness of A.
- But in general, it is *not* possible to construct a witness of A starting from a witness of not(not(A)).

Proof by Contradiction

Suppose we have a proposition P, of which we don't know whether it is true or false.

1. Assume the contrary, that is, suppose P is *false*.
2. Taking not(P) as a hypothesis, construct a proof of not(Q), where Q is a proposition which we know to be *true*.
3. Since *it is impossible to prove a false statement by starting from true hypotheses and reasoning correctly*, P cannot be false:
   By the law of excluded middle, it must be true.

Example: The square root of 2 is irrational

Claim

$\sqrt{2}$ is irrational.

Example: The square root of 2 is irrational

Claim

$\sqrt{2}$ is irrational.

Step 1: Assume the contrary.

- Suppose integers m and n exist such that $\sqrt{2} = \frac{m}{n}$.

Example: The square root of 2 is irrational

Claim

$\sqrt{2}$ is irrational.

Step 2: Construct a proof of a false statement.

- We may suppose $m, n \geq 1$ and $\gcd(m, n) = 1$.
- By squaring and multiplying by n^2 we get $m^2 = 2n^2$.
- As m^2 is even, so must be m.
- Let $m = 2k$. Then $4k^2 = 2n^2$, hence $2k^2 = n^2$.
- As n^2 is even, so must be n.
- So *m and n are two relatively prime integers, both even*.

Example: The square root of 2 is irrational

Claim

$\sqrt{2}$ is irrational.

Step 3: Conclude that the original proposition is true.

- If the square root of 2 is rational, then there are two relatively prime integers which are both even.
- But two relatively prime integers cannot be both even.
- Therefore, the square root of 2 cannot be rational: it must be irrational.

Proof by Contradiction vs Proof by Negation

Suppose we have a proposition P, of which we don't know whether it is true or false.

1. Suppose P is true.
2. Taking P as a hypothesis, construct a proof of not(Q), where Q is a predicate which we know to be true.
3. Since it is impossible to prove a false statement by starting from true hypotheses and reasoning correctly, P cannot be true: it must be false.

Proof by Contradiction vs Proof by Negation

Suppose we have a proposition P, of which we don't know whether it is true or false.

1. Suppose P is true.
2. Taking P as a hypothesis, construct a proof of not(Q), where Q is a predicate which we know to be true.
3. Since it is impossible to prove a false statement by starting from true hypotheses and reasoning correctly, P cannot be true: it must be false.

Is this the same kind of argument as proof by contradiction?

Proof by Contradiction vs Proof by Negation

Suppose we have a proposition P, of which we don't know whether it is true or false.

1. Suppose P is true.
2. Taking P as a hypothesis, construct a proof of not(Q), where Q is a predicate which we know to be true.
3. Since it is impossible to prove a false statement by starting from true hypotheses and reasoning correctly, P cannot be true: it must be false.

Is this the same kind of argument as proof by contradiction?

Yes and no:

- An argument by *contradiction* has the form:

  If not(A), then contradiction; thus, A.

- This new argument, however, has the form:

  If A, then contradiction; thus, not(A).

  This is more correctly called a *proof by negation*, rather than by contradiction.

- We *could* apply proof by negation to not(A), but we would get:

  If not(A), then contradiction; thus, not(not(A)).

- To conclude with A, we still need "if not(not(A)), then A": which is (another form of) the law of the excluded middle!

Next section

1 Proof by Contradiction

2 The Well Ordering Principle

3 Well Ordering Proofs

4 Factoring into Primes

5 Well Ordered Sets

6 Ambiguity with human language

7 Propositions from Propositions

8 Propositional Logic in Computer Programs

The Well Ordering Principle

Every *nonempty* set of *nonnegative* integers has a *smallest* element.

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. Propositional Logic in Computer Programs

Revisiting an old example

We saw a proof of the following:

Theorem

$\sqrt{2}$ is irrational.

At one point, the proof went:

- We may suppose $m, n \geq 1$ and $\gcd(m, n) = 1$.

Revisiting an old example

We saw a proof of the following:

Theorem

$\sqrt{2}$ is irrational.

At one point, the proof went:

- We may suppose $m, n \geq 1$ and $\gcd(m, n) = 1$.

Question: *why* could we suppose so?

Revisiting an old example

We saw a proof of the following:

Theorem

$\sqrt{2}$ is irrational.

At one point, the proof went:

- We may suppose $m, n \geq 1$ and $\gcd(m, n) = 1$.

Question: *why* could we suppose so?

Answer: because of the well ordering principle!

Every fraction can be written in lowest terms.

Suppose there exist positive integers m, n such that the fraction $\frac{m}{n}$ *cannot* be written in lowest terms.

- Let C be the set of those positive integers that are *numerators* of fractions which cannot be written in lowest terms.

Every fraction can be written in lowest terms.

Suppose there exist positive integers m, n such that the fraction $\frac{m}{n}$ *cannot* be written in lowest terms.

- Let C be the set of those positive integers that are *numerators* of fractions which cannot be written in lowest terms.
- Then C is nonempty, because it contains m:
- Let m_0 be the smallest element of C.
- Correspondingly, let n_0 be such that $\frac{m_0}{n_0}$ cannot be written in lowest terms.
- Then m_0 and n_0 must have a *common prime factor* p:
  Otherwise, $\frac{m_0}{n_0}$ would be a writing in lower terms.

Every fraction can be written in lowest terms.

Suppose there exist positive integers m, n such that the fraction $\frac{m}{n}$ *cannot* be written in lowest terms.

- Let C be the set of those positive integers that are *numerators* of fractions which cannot be written in lowest terms.
- We have established the following:

  If m_0 is the smallest element of C,
  and $\frac{m_0}{n_0}$ cannot be written in lowest terms,
  then m_0 and n_0 have a common prime factor p.

- But $\frac{m_0/p}{n_0/p} = \frac{m_0}{n_0}$, so $\frac{m_0}{p}$ must also belong to C.
- But this is impossible, because $\frac{m_0}{p} < m_0$, and m_0 is the smallest element of C.

Notation for sets

Let $P(x)$ be a predicate whose truth value depends on the value of variable x.

- If an object x is in a set S, we write: $x \in S$.
- We denote by:

  $$\{x \in S \mid P(x)\}$$

  the set of *all and only* those elements x of S for which $P(x)$ is true.

  - We read: "the set of the x in S such that $P(x)$".
  - We may omit S if irrelevant or clear from the context.
- If $E(x)$ is an expression that involves x, we can use the abbreviation:

  $$\{E(x) \mid P(x)\} ::= \{y \mid \text{there exists } x \text{ such that } P(x) \text{ and } y = E(x)\}$$

  We read: "the set of the $E(x)$ where $P(x)$".
- The *empty set* which has no elements at all is denoted by $\emptyset$.

A template for well ordering proofs

Call $\mathbb{N}$ the set of the *nonnegative* integers.
Let $P(n)$ be a predicate which depends on a variable n taking values in $\mathbb{N}$.
We want to prove that $P(n)$ is true for *every* $n \in \mathbb{N}$.

1. Let C be the set of *counterexamples*:

$$C = \{c \in \mathbb{N} \mid P(c) \text{ is false}\}$$

2. By contradiction, assume that C is nonempty.
3. By the Well Ordering Principle, C has a smallest element c_0:
   This c_0 is the *smallest counterexample*.
4. Derive a contradiction. Some ways to do so:
   - Show that $P(c_0)$ is true: that is, c_0 is *not a counterexample*.
   - Show that C has an element c_1 smaller than c_0:
     that is, c_0 is *not the smallest* counterexample.
   - Use c_0 to construct a proof of not(Q) where Q is a proposition which is known to be true.
5. Conclude that C is empty, hence $P(n)$ is true for every $n \in \mathbb{N}$.

Notation for sums

Let $a \in \mathbb{N}$ be fixed and, for an integer $n \geq a$ and all integers k such that $a \leq k \leq n$, let x_k be a number.

The *sum, for k from a to n, of the x_k* is the number $\sum_{k=a}^{n} x_k$ defined as follows:

- If $n = a$, then $\sum_{k=a}^{a} x_k = x_a$.
- If $n > a$, then $\sum_{k=a}^{n} x_k = \left(\sum_{k=a}^{n-1} x_k\right) + x_n$.

This is our first example of a *recursive definition*, where a *base case* is given, and every next object can be obtained from the previous ones by applying one or more *constructors*.

Example: the sum of the first n positive integers

Theorem

For every positive integer n, $1+2+\ldots+n=\dfrac{n(n+1)}{2}$.

The notation on the left-hand side means: the sum of all positive integers from 1 to n.

- Let $C=\left\{c\in\mathbb{N}\mid c>0 \text{ and } 1+2+\ldots+c\neq\dfrac{c(c+1)}{2}\right\}$.
- If C is nonempty, then it has a smallest element c_0.
- We observe that c_0 cannot be 1, because $1=\dfrac{1\cdot 2}{2}$.
- Then c_0-1 is a positive integer, and as it is smaller than c_0,

$$1+2+\ldots+(c_0-1)=\frac{(c_0-1)c_0}{2}\,.$$

- But then,

$$1+2+\ldots+c_0=\frac{(c_0-1)c_0+2c_0}{2}=\frac{c_0(c_0+1)}{2}\ :\ \text{contradiction}\,.$$

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. Propositional Logic in Computer Programs

The Prime Factorization Theorem

An integer $p \geq 2$ is *prime* if its only positive divisors are 1 and p itself.

Theorem 2.3.1.

Every integer $n \geq 2$ can be factored as a product of primes.

The Prime Factorization Theorem

An integer $p \geq 2$ is *prime* if its only positive divisors are 1 and p itself.

Theorem 2.3.1.

Every integer $n \geq 2$ can be factored as a product of primes.

Proof: by the Well Ordering Principle.

- Let C be the set of counterexamples to Theorem 2.3.1, that is, the integers $n \geq 2$ which *cannot* be factored as a product of primes.
- By contradiction, assume that C is nonempty.

The Prime Factorization Theorem

An integer $p \geq 2$ is *prime* if its only positive divisors are 1 and p itself.

Theorem 2.3.1.

Every integer $n \geq 2$ can be factored as a product of primes.

Proof: by the Well Ordering Principle.

- Let C be the set of counterexamples to Theorem 2.3.1, that is, the integers $n \geq 2$ which *cannot* be factored as a product of primes.
- By contradiction, assume that C is nonempty.
- Let c_0 be the least element of C, as ensured by the Well Ordering Principle.
- Then c_0 cannot be prime, because a *product of one prime* is still a product of primes.
- Then c_0 has a positive divisor a such that $1 < a < c_0$.
- But then, $b = c_0/a$ is also such that $1 < b < c_0$.

The Prime Factorization Theorem

An integer $p \geq 2$ is *prime* if its only positive divisors are 1 and p itself.

Theorem 2.3.1.

Every integer $n \geq 2$ can be factored as a product of primes.

Proof: by the Well Ordering Principle.

- Let C be the set of counterexamples to Theorem 2.3.1, that is, the integers $n \geq 2$ which *cannot* be factored as a product of primes.
- By contradiction, assume that C is nonempty.
- Then the least element c_0 of C satisfies $c_0 = a \cdot b$ where $1 < a < c_0$ and $1 < b < c_0$.
- But as a and b are smaller than c_0, and c_0 is the smallest counterexample, a and b *can* be written as products of primes!
- So let $a = p_1 p_2 \cdots p_m$ and $b = q_1 q_2 \cdots q_n$ be writings of a and b as products of primes.
- Then $ab = p_1 p_2 \cdots p_m q_1 q_2 \cdots q_n$ is a writing of c_0 as a product of primes!

The Prime Factorization Theorem

An integer $p \geq 2$ is *prime* if its only positive divisors are 1 and p itself.

Theorem 2.3.1.

Every integer $n \geq 2$ can be factored as a product of primes.

Proof: by the Well Ordering Principle.

- Let C be the set of counterexamples to Theorem 2.3.1, that is, the integers $n \geq 2$ which *cannot* be factored as a product of primes.
- By contradiction, assume that C is nonempty.
- We can summarize our findings as follows:

  If there are any counterexamples to Theorem 2.3.1,
  then the smallest such counterexample is not a counterexample.

- This is impossible, so there was no counterexample in the first place, and Theorem 2.3.1 is true.

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. Propositional Logic in Computer Programs

Well ordered sets

Definition

A set S of numbers is *well ordered* if every nonempty subset of S has a minimum element.

Well ordered sets

Definition

A set S of numbers is *well ordered* if every nonempty subset of S has a minimum element.

The Well Ordering Principle can then be restated as follows:

The set of nonnegative integers is well ordered.

Are there other well ordered sets? Indeed:

Every *nonempty* subset of a well ordered set is well ordered.

A small, but useful, generalization

Theorem

For every $n \in \mathbb{N}$ the set of integers *no smaller than $-n$* is well ordered.

A small, but useful, generalization

Theorem

For every $n \in \mathbb{N}$ the set of integers *no smaller than* $-n$ is well ordered.

We give an argument which does not depend on the specific value of n, hence holds for every n.

- Let S be a nonempty set of integers no smaller than $-n$.
- Then $S+n = \{m+n \mid m \in S\} \subseteq \mathbb{N}$.
- By the Well Ordering Principle, $S+n$ has a minimum m_0.
- Then $m_0 - n$ is the minimum of S.

Two quick corollaries

Definition

A *lower bound* (resp., *upper bound*) for a set S of real numbers is a real number b such that $b \leq s$ (resp., $b \geq s$) for every $s \in S$.

Corollary 1

Any nonempty set of integers with a lower bound is well ordered.

Proof: If b is a lower bound for S, and $n \geq |b|$, then S is a subset of $\{x \in \mathbb{Z} \mid x \geq -n\}$, which is well ordered.
Here, $|x| = \max(x, -x)$ is the *absolute value* of x.

Corollary 2

Any nonempty set of integers with an upper bound has a greatest element.

Proof:

- If b is an *upper* bound for S, then $-b$ is a *lower* bound for $-S = \{-s \mid s \in S\}$.
- If m is the smallest element of $-S$, then $-m$ is the greatest element of S.

Another example

Let $\mathbb{F} = \left\{ \frac{n}{n+1} \mid n \in \mathbb{N} \right\}$.

Lemma (2.4.7 on textbook)

$\mathbb{F}$ is well ordered.

Proof:

- Let $S \subseteq \mathbb{F}$ be nonempty.
- Let $n_0 = \min \left\{ n \in \mathbb{N} \mid \frac{n}{n+1} \in S \right\}$.
- Then $m = \frac{n_0}{n_0+1}$ is the minimum of S.

  This is more easily seen by writing $\frac{n}{n+1}$ as $1 - \frac{1}{n+1}$.

Another example

Let $\mathbb{F} = \left\{ \frac{n}{n+1} \mid n \in \mathbb{N} \right\}$.

Theorem

$\mathbb{N} + \mathbb{F} = \{ n + f \mid n \in \mathbb{N}, f \in \mathbb{F} \}$ is well ordered.

Another example

Let $\mathbb{F} = \left\{ \frac{n}{n+1} \mid n \in \mathbb{N} \right\}$.

Theorem

$\mathbb{N} + \mathbb{F} = \{ n + f \mid n \in \mathbb{N}, f \in \mathbb{F} \}$ is well ordered.

Before we prove this, think: *why so?*

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. Propositional Logic in Computer Programs

An issue with human language

Consider these statements:

1. You can have the cake, or eat it.
2. If two and two are five, then I am the Pope.
3. If you can solve any exercise, then you will pass the test.
4. Everyone has a dream.

What do they *mean*? It might not be immediately clear.
Which is not surprising, because:

Human language is ambiguous.

This is fine: or we must renounce to poetry, humour, etc.
But it is inconvenient when we do mathematics . . .

An issue with human language

Consider these statements:

1. You can have the cake, or eat it.
2. If two and two are five, then I am the Pope.
3. If you can solve any exercise, then you will pass the test.
4. Everyone has a dream.

What do they *mean*? It might not be immediately clear.
Which is not surprising, because:

Human language is ambiguous.

This is fine: or we must renounce to poetry, humour, etc.
But it is inconvenient when we do mathematics ...

Ambiguity: inclusion and exclusion

You can have the cake, or eat it.

- Can I have the cake and also eat it?
- Must I renounce to eat the cake if I want to have it?

Ambiguity: false hypotheses, true consequences

If two and two are five, then I am the Pope.

- What if I am not the Pope?
- What if I *am* the Pope?
- What if two and two are actually five, but I am not the Pope?

Ambiguity: “some” vs “all”

If you can solve any exercise, then you will pass the test.

- Can I pass the test if I solve only one exercise?
- Do I need to solve an exercise in particular?
- Do I need to solve every single exercise?

Ambiguity: “for every” vs “exists”

Everyone has a dream.

- Does every single person have a dream of their own?
- Is there a single dream that everyone has?

A non-ambiguous language for mathematics

To avoid ambiguities, mathematicians divide propositions into *atomic formulas* joined together by *logical connectives*.

The role of atomic formulas is taken by *propositional variables* which can take any of the two values T (true) and F (false).

The relation between the truth values of the variables and that of a formula can be expressed by *truth tables*.

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. **Propositions from Propositions**
8. Propositional Logic in Computer Programs

The connective not$(\cdot)$

Truth value of not$(\cdot)$

If P is a proposition, then not(P) is also a proposition.
not(P) is true iff P is false.

The connective not$(\cdot)$ is also called *negation*.

Truth table for not$(\cdot)$

P	not(P)
T	F
F	T

The connective and

Truth value of P and Q

If P and Q are propositions, then P and Q is also a proposition. P and Q is true iff both P and Q are true.

The connective and is also called *conjunction*.

Truth table for and

P	Q	P and Q
T	T	T
T	F	F
F	T	F
F	F	F

The connective or

Truth value of P or Q

If P and Q are propositions, then P or Q is also a proposition.
P or Q is true iff either P or Q is true, or both are.

The connective or is also called *disjunction*.

Truth table for or

P	Q	P or Q
T	T	T
T	F	T
F	T	T
F	F	F

The connective xor

Truth value of P xor Q

If P and Q are propositions, then P xor Q is also a proposition. P xor Q is true iff either P or Q is true, but not both.

xor (ex-OR) is also called *exclusive or*, or *exclusive disjunction*.

Truth table for xor

P	Q	P xor Q
T	T	F
T	F	T
F	T	T
F	F	F

You can have the cake, or eat it

Let P be the proposition "I can have the cake", and Q be the proposition "I can eat the cake".

Can I have the cake and also eat it?

This corresponds to P or Q.

Do I lose the cake if I eat it?

This corresponds to P xor Q.

The connective implies

Truth value of P implies Q

If P and Q are propositions, then P implies Q is also a proposition. P implies Q is true iff either P is false, or Q is true.

P implies Q can be read as follows:

- If P, then Q.
- P is a *sufficient* condition for Q.
- Q is a *necessary* condition for P.

Truth table for implies

P	Q	P implies Q
T	T	T
T	F	F
F	T	T
F	F	T

The connective implies

Truth value of P implies Q

If P and Q are propositions, then P implies Q is also a proposition. P implies Q is true iff either P is false, or Q is true.

This is called the *material implication:*

> "P implies Q" means "it is never the case that P without Q".

Important: it is *not* necessary that P be a *cause* for Q!

Truth table for implies

P	Q	P implies Q
T	T	T
T	F	F
F	T	T
F	F	T

If two and two are five, then I am the Pope

Let P be the proposition “two and two are five”, and Q be the proposition “I am the Pope”.

What if I am not the Pope?

Anyway, P implies Q has a false antecedent, so it is true.

What if I **am** the Pope?

Then P implies Q has a true consequent, so it is true.

What if two and two are actually five, but I am not the Pope?

Then P implies Q would have a true antecedent, and a false consequent: so it would be false.

The connective iff

Truth value of P iff Q

If P and Q are propositions, then P iff Q is also a proposition.
P iff Q is true iff P and Q are either both true, or both false.

That is:

“P iff Q” means “P and Q have the same truth value”.

Truth table for iff

P	Q	P iff Q
T	T	T
T	F	F
F	T	F
F	F	T

Next section

1. Proof by Contradiction
2. The Well Ordering Principle
3. Well Ordering Proofs
4. Factoring into Primes
5. Well Ordered Sets
6. Ambiguity with human language
7. Propositions from Propositions
8. **Propositional Logic in Computer Programs**

Condition checking with propositional logic

Consider a piece of Python code such as:

```
if x > 0 or (x <= 0 and y > 100):
    %% your code here
```

- Can we determine if and when your code will be run?
- Can we write the if-condition in a simpler form?

Let us consider the following propositions:

- $A ::= \mathtt{x\ >\ 0}$
- $B ::= \mathtt{y\ >\ 100}$

We observe that $\mathtt{x} <= 0$ is just not(A), so:

$$\mathtt{x\ >\ 0\ \ or\ \ (x\ <=\ 0\ \ and\ \ y\ >\ 100)}$$
$$\text{corresponds to } A \text{ or } (\text{not}(A) \text{ and } B)$$

Condition checking with propositional logic

Consider a piece of Python code such as:

```
    if x > 0 or (x <= 0 and y > 100):
        %% your code here
```

- Can we determine if and when your code will be run?
- Can we write the if-condition in a simpler form?

Let us consider the following propositions:

- $A ::= \texttt{x} > 0$
- $B ::= \texttt{y} > 100$

We observe that $\texttt{x} <= 0$ is just not(A), so:

$$\texttt{x > 0 or (x <= 0 and y > 100)}$$

$$\text{corresponds to } A \text{ or } (\text{not}(A) \text{ and } B)$$

Equivalent formulas

Definition

Let α and β be formulas in the variables $P_1, \ldots, P_n$.
α and β are *equivalent* if each assignment of truth values to $P_1, \ldots, P_n$ makes α and β either both true, or both false.

Equivalent formulas

Definition

Let α and β be formulas in the variables $P_1, \ldots, P_n$.
α and β are *equivalent* if each assignment of truth values to $P_1, \ldots, P_n$ makes α and β either both true, or both false.

Examples:

- $\alpha ::= P \text{ or } Q$ and $\beta ::= \text{not}(\text{not}(P) \text{ and not}(Q))$.
- $\alpha ::= P \text{ implies } (Q \text{ implies } P)$ and $\beta ::= R \text{ or not}(R)$.

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

We start with the basics of the table:

A	B	A or	(not(A)	and B)	A or B
T	T				
T	F				
F	T				
F	F				

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

We fill the rightmost column, and take a note of the values:

A	B	A or	(not(A)	and B)	A or B
T	T				T
T	F				T
F	T				T
F	F				F

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

We convert A into not(A), and take note of the values:

A	B	A or	(not(A)	and B)	A or B
T	T		F		T
T	F		F		T
F	T		T		T
F	F		T		F

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

We now determine the values of (not(A) and B):

A	B	A or	(not(A)	and B)	A or B
T	T		F	F	T
T	F		F	F	T
F	T		T	T	T
F	F		T	F	F

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

Finally, we determine the values of A or (not(A) and B):

A	B	A or	(not(A)	and B)	A or B
T	T	T	F	F	T
T	F	T	F	F	T
F	T	T	T	T	T
F	F	F	T	F	F

Truth table calculation

Claim

A or (not(A) and B) is equivalent to A or B.

Finally, we determine the values of A or (not(A) and B):

A	B	A or	(not(A)	and B)	A or B
T	T	T	F	F	T
T	F	T	F	F	T
F	T	T	T	T	T
F	F	F	T	F	F

. . . and we see that they always match, proving the claim.

We can then rewrite the snippet as:

```
if x > 0 or y > 100:
    %% your code here
```

Simplifying by reasoning

We can also prove the equivalence by reasoning case by case:
(and making some observations in the meantime)

$A = \mathrm{T}$ A formula of the form T or Q has truth value T.
If A is T, so are both A or (not(A) and B) and A or B.

$A = \mathrm{F}$ A formula of the form F or Q, or of the form T and Q, has the same truth value as Q.
If A is F, then not(A) and B has the same truth value of B, and so do A or (not(A) and B) and A or B.

In either case, A or (not(A) and B) and A or B take the same truth value on each assignment of A and B.

Why simplify?

1. To improve *readability*.
   Conditions with a simple structure are more easily checked than complex ones.
2. To increase *speed*.
   Less complex formulas require less time to be evaluated.
3. To reduce *cost*.
   The formula might refer to a circuit, whose realization requires materials, tools, time, and money.

Symbolic notation for logical connectives

English	Symbolic
not(P)	$\neg P, \overline{P}$
P and Q	$P \wedge Q$
P or Q	$P \vee Q$
P xor Q	$P \oplus Q$
P implies Q	$P \longrightarrow Q$
P iff Q	$P \longleftrightarrow Q$

Precedence

From strongest to weakest:

1. not(·)
2. and
3. or
4. xor
5. implies
6. iff

For example,

$$\text{not}(A) \text{ and } B \text{ or } C \text{ implies } D \text{ iff } E \text{ xor } F$$

is a shortcut for

$$((((\text{not}(A)) \text{ and } B) \text{ or } C) \text{ implies } D) \text{ iff } (E \text{ xor } F)$$

When in doubt: use parentheses.